COURSE: COMPUTER NETWORKS LABORATORY

**CSN-361**

REPORT ON

ASSIGNMENT 3

**Name: Deepansh Nagaria**

**Enrollment Number: 17114024**

**CSE 3rd YEAR**

**PROBLEM STATEMENT 1**

**Write a socket program in C to determine class, Network and Host ID of an IPv4 address.**

**ALGORITHMS USED :**

* **For determining the class:** The idea is to check first octet of IP address. As we know, for class **A**first octet will range from **1 – 126**, for class **B** first octet will range from **128 – 191**, for class **C** first octet will range from **192- 223**, for class **D** first octet will range from **224 – 239**, for class **E** first octet will range from **240 – 255**.
* **For determining the Network and Host ID:** We know that [Subnet Mask](https://www.iplocation.net/subnet-mask) for Class **A** is **8**, for Class **B**is **16** and for Class **C** is **24** whereas Class **D** and **E** is not divided into Network and Host ID.
* For 2nd Example, first octet is 130. So, it belongs to Class **B**. Class B has subnet mask of 16. So, first 16 bit or first two octet is Network ID part and rest is Host ID part.  
  Hence, Network ID is **130.45** and Host ID is **151.154**
* Various **Function Calls** used are:

1. NetworkClass()
2. HostandNetwork\_ID : Function to determine the network ID and Host ID of ip address.

**DATA STRUCTURES/FUNCTIONS USED :**

* Primitive data types(int, char, etc.).
* Character arrays(Example: str[], arr[], etc.)

**SNAPSHOTS:**

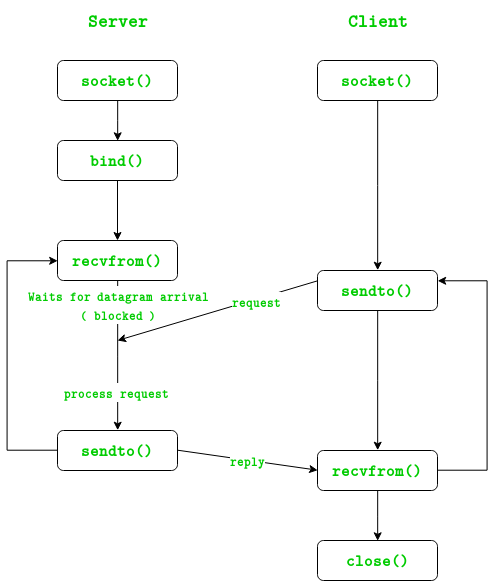
**A screen shot of a computer

Description automatically generated**

**PROBLEM STATEMENT 2**

**Write a C program to demonstrate File Transfer using UDP.**

**ALGORITHMS USED :**



* The server starts and waits for filename.
* The client sends a filename.
* The server receives filename. If file is present, server starts reading file and continues to send a buffer filled with file contents encrypted until file-end is reached.
* End is marked by EOF.
* File is received as buffers until EOF is received. Then it is decrypted.
* If Not present, a file not found is sent.

**DATA STRUCTURES USED :**

* **Int, char \*, char []:** To store the socket , strings, buffer
* **struct sockaddr\_in :** struct sockaddr\_in is the structure used with IPv4 addresses (e.g. "192.0.2.10"). It holds an address family (AF\_INET), a port in sin\_port, and an IPv4 address in sin\_addr.

**SNAPSHOTS:**

A screen shot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

**PROBLEM STATEMENT 3**

**Write a TCL code for network simulator NS2 to demonstrate the star topology among a set of computer nodes. Given N nodes, one node will be assigned as the central node and the other nodes will be connected to it to form the star. You have to set up a TCP connection between k pairs of nodes and demonstrate the packet transfer between them using Network Animator (NAM). Use File Transfer protocol (FTP) for the same. Each link should have different color of packets to differentiate the packets transferred between each pair of nodes. The program should take the number of nodes (N) as input followed by k pairs of nodes.**

**ALGORITHMS USED :**

* + Input the number of nodes (N).
  + Create n nodes of network and connect them using a duplex-link
  + Then input k (number of pairs for TCP connection)
  + Then input the pairs.
  + For each pair create a TCP connection using TCP Agent and sink Agent
  + Then specify the FTP porotcol to be used for transmission of data.
  + Specify the events along with their time
  + Finally finish the program by calling the finish process

**DATA STRUCTURES USED :**

**In-Built structures**:

1. **set ns [new Simulator]**
2. **set n($i) [$ns node]**
3. **set tcp($i) [new Agent/TCP] :** TCP agent to establish TCP connection staring node.
4. **set sink($i) [new Agent/TCPSink] :** TCP sink agent to establish TCP connection sink node.
5. **set ftp($i) [new Application/FTP] :** FTP object to specify FTP protocol on TCP connection

Various **Algorithms/Function Calls** used are:

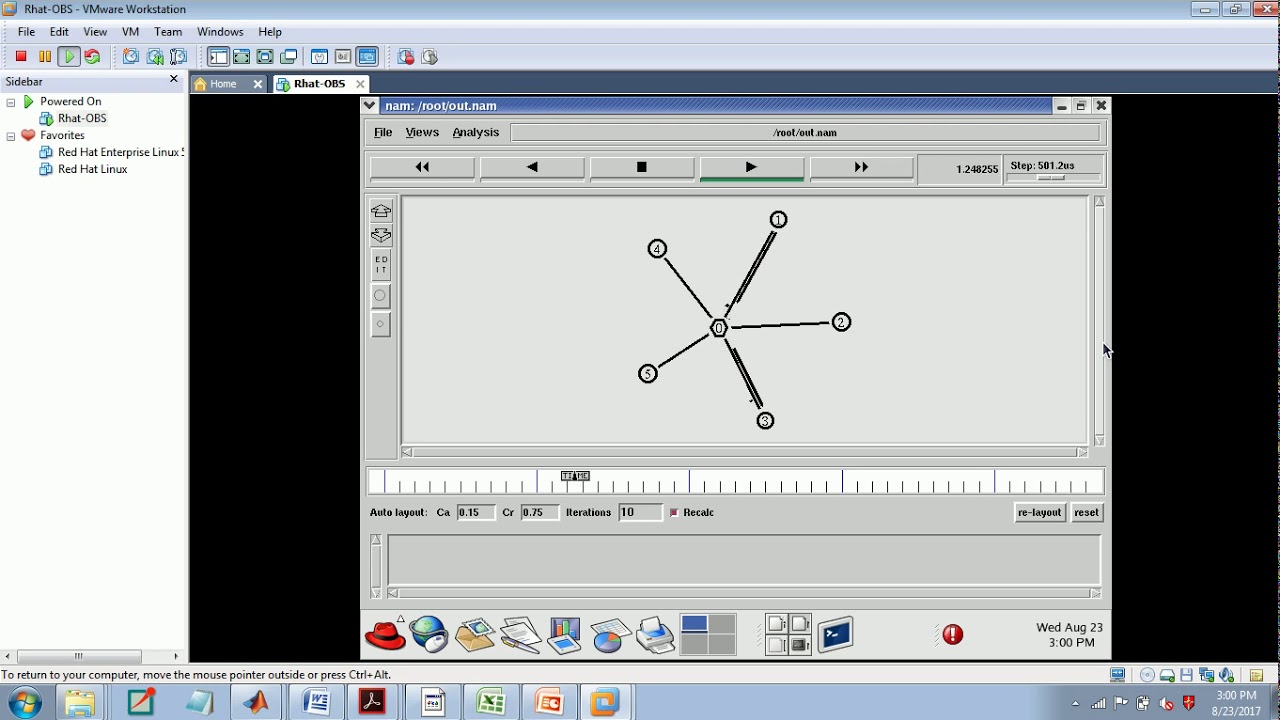
1. **proc finsh():**

Function to specify what to do when the program has ended executing.

1. **$ns duplex-link $n(0) $n($i) 512Kb 10ms SFQ**
   * + To create a double ended link between two network nodes and specify its details like delay, transmission speed, how to manage the queue.
2. **$ns attach-agent $n($n1($i)) $tcp($i) :** 
   * + To attach the specified agent to the specified node in the network
3. **$ns connect $tcp($i) $sink($i)**

To connect the two nodes to form a connection between them as specified (tcp, udp, etc)

**SNAPSHOTS:**



**PROBLEM STATEMENT 4**

**Write a TCL code for network simulator NS2 to demonstrate the ring topology among a set of computer nodes. Given N nodes, each node will be connected to two other nodes in the form of a ring. You have to set up a TCP connection between k pairs of nodes and demonstrate packet transfer between them using Network Animator (NAM). Use File Transfer protocol (FTP) for the same. Each link should have different color of packets to differentiate the packets transferred between each pair of nodes. The program should take the number of nodes (N) as input followed by k pairs of nodes.**

**ALGORITHMS USED :**

* + Input the number of nodes (N).
  + Create n nodes of network and connect them using a duplex-link
  + Then input k (number of pairs for TCP connection)
  + Then input the pairs.
  + For each pair create a TCP connection using TCP Agent and sink Agent
  + Then specify the FTP porotcol to be used for transmission of data.
  + Specify the events along with their time
  + Finally finish the program by calling the finish process

**DATA STRUCTURES USED :**

**In-Built structures**:

1. **set ns [new Simulator]**
2. **set n($i) [$ns node]**
3. **set tcp($i) [new Agent/TCP] :** TCP agent to establish TCP connection staring node.
4. **set sink($i) [new Agent/TCPSink] :** TCP sink agent to establish TCP connection sink node.
5. **set ftp($i) [new Application/FTP] :** FTP object to specify FTP protocol on TCP connection

Various **Algorithms/Function Calls** used are:

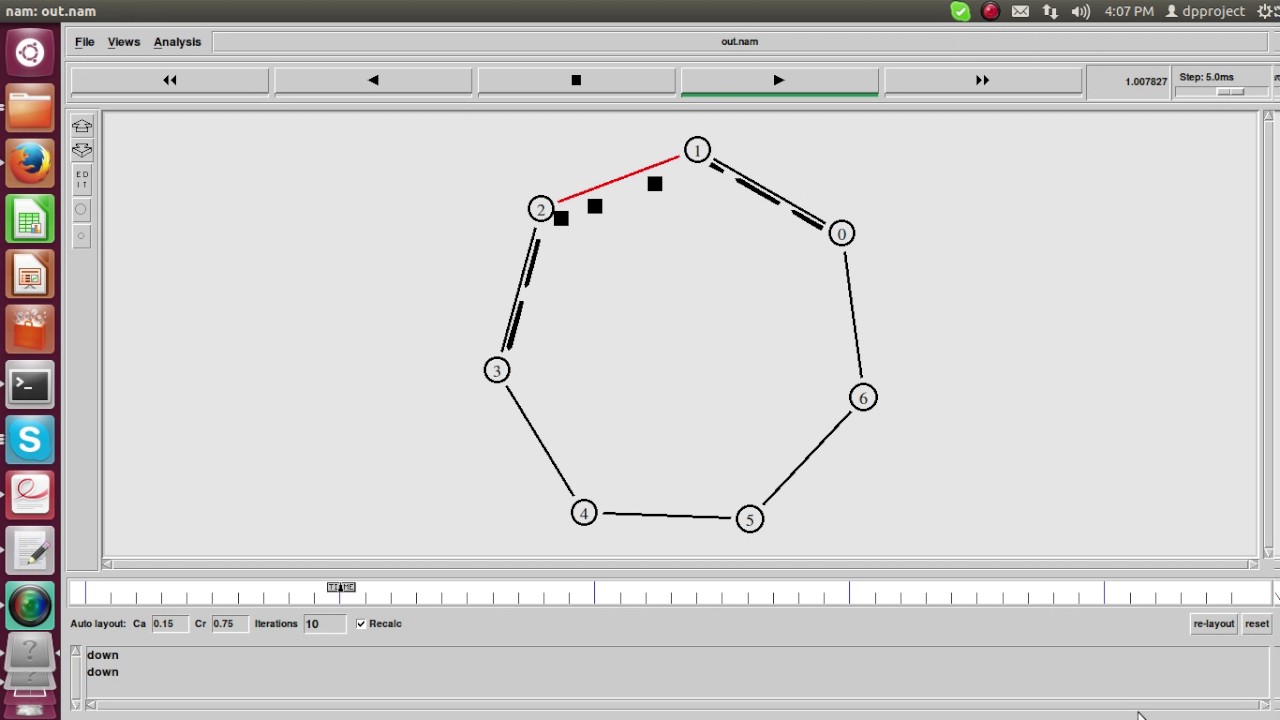
1. **proc finsh():**

Function to specify what to do when the program has ended executing.

1. **$ns duplex-link $n(0) $n($i) 512Kb 10ms SFQ**
   * + To create a double ended link between two network nodes and specify its details like delay, transmission speed, how to manage the queue.
2. **$ns attach-agent $n($n1($i)) $tcp($i) :** 
   * + To attach the specified agent to the specified node in the network
3. **$ns connect $tcp($i) $sink($i)**

To connect the two nodes to form a connection between them as specified (tcp, udp, etc)

**SNAPSHOTS:**



**PROBLEM STATEMENT 5**

**Write a TCL code for network simulator NS2 to demonstrate the bus topology among a set of computer nodes. Given N nodes, each node will be connected to two other nodes in the form of a ring. You have to set up a TCP connection between k pairs of nodes and demonstrate packet transfer between them using Network Animator (NAM). Use File Transfer protocol (FTP) for the same. Each link should have different color of packets to differentiate the packets transferred between each pair of nodes. The program should take the number of nodes (N) as input followed by k pairs of nodes.**

**ALGORITHMS USED :**

* + Input the number of nodes (N).
  + Create n nodes of network and connect them using a duplex-link
  + Then input k (number of pairs for TCP connection)
  + Then input the pairs.
  + For each pair create a TCP connection using TCP Agent and sink Agent
  + Then specify the FTP porotcol to be used for transmission of data.
  + Specify the events along with their time
  + Finally finish the program by calling the finish process

**DATA STRUCTURES USED :**

**In-Built structures**:

1. **set ns [new Simulator]**
2. **set n($i) [$ns node]**
3. **set tcp($i) [new Agent/TCP] :** TCP agent to establish TCP connection staring node.
4. **set sink($i) [new Agent/TCPSink] :** TCP sink agent to establish TCP connection sink node.
5. **set ftp($i) [new Application/FTP] :** FTP object to specify FTP protocol on TCP connection

Various **Algorithms/Function Calls** used are:

1. **proc finsh():**

Function to specify what to do when the program has ended executing.

1. **$ns duplex-link $n(0) $n($i) 512Kb 10ms SFQ**
   * + To create a double ended link between two network nodes and specify its details like delay, transmission speed, how to manage the queue.
2. **$ns attach-agent $n($n1($i)) $tcp($i) :** 
   * + To attach the specified agent to the specified node in the network
3. **$ns connect $tcp($i) $sink($i)**

To connect the two nodes to form a connection between them as specified (tcp, udp, etc)

**SNAPSHOTS:**

